Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 132 (2018) 1041–1048

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Computational Intelligence and Data Science (ICCIDS 2018)

Encoder-Decoder Architectures for Generating Questions

Jaspreet Singh[a], Yashvardhan Sharma[a]

[a] *Department of Computer Science Information System*
Birla Institute of Technology Science, Pilani Campus Pilani-333031

Abstract

With exploding textual data on the internet with e-books, legal documents and products information, it is an opportunity to harness it for applications which can aid human tasks. Developing systems for question generation can be used for making frequently-asked-questions, creating school quiz-es and serve for the purpose of unified AI. Here in this study various encoder decoder architectures for generating questions from text inputs have been explored using Stanford's SQuAD dataset as for training development and test sets and evaluation metrics such as BLEU, ROUGUE and training time were used to compare the effectiveness of the models. The article develops upon the work of current end-to-end system by using gated recurrent unit in place of long short term memory which give similar accuracy but with lesser training time, further it also show the successfully use of a convolution based encoder for this task which gives results comparable to current state of the art system with much lesser training time.

Peer-review under responsibility of the scientific committee of the International Conference on Computational Intelligence and Data Science (ICCIDS 2018).

Keywords: Automatic Question Generation; Neural Networks; Language Generation; Natural Language Processing

1. Introduction

Automatic Question generation is an artificial intelligence task in which the objective is to obtain interrogative text from system which inquires about input sentence or paragraph. Question Generation systems have many plausible real life application once they achieve reasonable performance they can be incorporated with chat bot interfaces to improve user experience or be used in educational environments to create vocabulary quiz [2], reading comprehension assessment [4] [5] or contribute to feedback systems for writing enhancement [3]. Possibly improvement in this domain can also contribute to development of machine interfaces to treat mental health [6].Further popularization of MOOCs have made automatic assessment a necessity to minimize human involvement and conduct such projects in a viable manner. Previous attempt for question generation have been by application of rule based techniques which made linguistic transformation to create a interrogative form of the input text [4, 7]. In 2010 Heilman[8] created a

* Corresponding author. Tel.: +91-992-995-7685
E-mail address: f2014152@pilani.bits-pilani.ac.in

1877-0509
Peer-review under responsibility of the scientific committee of the International Conference on Computational Intelligence and Data Science (ICCIDS 2018).
10.1016/j.procs.2018.05.019

system which generated questions based on linguistic rules but were also ranked using trained(supervised learning) ranker for question scoring. But the development of better datasets like SQuAD [1] and Ms macro [9] led to using of more data driven approaches which are described in the following section of related work. In this work analysis has been made on various neural network models predominantly based on sequence to sequence learning paradigm which learn mapping between inputs and output sequence, without the use of any hand crafted rule in a statistical manner. On a high level the article describes experiments on designing of recurrent encoder with use of LSTMs, GRUs and convolutional encoder for this task.
The proceeding sections describe previously done work in this domain, a mathematical formulation of the problem statement, description of the dataset used, the architecture of models developed and setup details of experiments. The last two section contains the results their its analysis and conclusions that can be drawn from presented work.

2. Related Work

One of the earliest work in question generation by Mitkov and Ha (2003) [4] used techniques based on syntactic methods to generate questions and used WordNet [10] to add semantic distractors. Mostow et al.[16] in 2009 used template based method to convert the parsed syntactic representation into question form. One of the foremost and successful machine learning based system was developed by Heilman and Smit[8] in 2010 which over generated questions from a piece of text then used a ranking algorithms, trained in a supervised way to score questions.The works of Agarwal et al. in 2011[11], Becker et al. in 2012 [12] use a different approach for this task by creating fill-in-the-blank questions. A single sentence is converted to multiple choice question using a gap and rest of the grammar intact.In 2013 Lindberg et al.[15] and Mazidi et al.[13] in 2014 developed methods of extracting semantic role patterns to form questions. Labutov et al. in 2015 [14] used ontological techniques to form document representation and crowd sourcing to gather data for relevance classification.
All these previous generation of works failed to capture semantic and diversity which natural language challenges throw in real life application.Thus more recent work on question generation have taken on more data driven approaches, first paper presenting end to end neural model used also encoder decoder with attention [17], the results on this model were further improved by researcher at Maluuba and MILA [18] using reinforcement learning, thus training a better policy gradient to optimize. Wang et al. [19] have created a model which jointly learns over the task of question generation as well as answering.
Here in this article work done in [17] is extended, by experimenting it with gated recurrent units and the use of convolutional encoder,thereafter analyzing the architectures over various hyper-parameters and design choices.

3. Problem Description

The mathematical formulation of the question generation task is described below, being modelled as a sequence to sequence learning task it will have a input sentence s which should mapped to a question sequence q each having a arbitrary length of $|s|$ and $|q|$ respectively. Basically the formulation tries to optimize the model to generate maximum probability of output sequence for a given input stream.

$$\bar{q} = \underset{q}{\operatorname{argmax}} P(q|s)$$

4. Dataset

SQuAD dataset[1] was used for conducting the experiments.It has been created using 23,215 paragraphs belonging to 536 Wikipedia articles by asking crowd sourced workers to create and answer question on each paragraph given to them. The dataset is distributed in JSON format with paragraph corresponding to question answer pairs. The creators of the dataset have kept the test set of the data hidden form public. Thus a training(80%), development(10%) and test set(10%) split for experimental purposes have been created.The the preprocessing of the data has been done using a similar approach as in [17], by sentence splitting the whole data and then lower casing it. After taht question answer

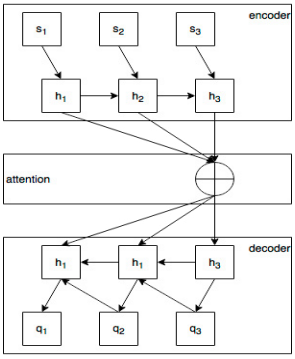

Fig. 1. A simple encoder decoder with attention

Table 1. Dataset Statistics

Data Info	Count
Training pairs	69,979
Development pairs	8,747
Testing pairs	8,747
Source vocab	45,004
Target vocab	28,003

pair were created, where the whole sentence(s) containing the annotated answer is taken. The set of sentence act as source sequence and questions are desired target sequence.Some more statistic about the data can seen in Table 1.

5. Model

The model being used to solve question generation is variant or development on the ideas laid by [21] and [22] and also deploys global attention [23] with the encoder used below. On a high level the system works by encoding input sentence into hidden representation then attention is applied to these hidden representation and further both these computed values are passed to decoder to make prediction optimizing the probability function described in section 3. The approach of answer to sentence generation is used without taking in to account the context of related paragraph, as described in the study [17] the former give better results. The possible explanation for this observation is additional information makes the model more complex and reduces the focus on answer sequence. The Fig. 1 describes skeleton for a recurrent sequence to sequence model which would be applicable to first LSTM and GRU based encoders.

5.1. Encoder

In this section the architecture of each of the encoder used has been described and in proceeding sections experimental settings and results for each variant are discussed. Let us assume the input sequence be in the form of $(s_1, s_2, ..., s_{|s|})$, an encoder should compute the corresponding sequence of hidden state or multiple stacked hidden states like $(h_1, h_2, ..., h_{|t|})$, which will then be passed on to decoder.

5.1.1. LSTM

Long Short Term Memory Networks were introduced by Hochreiter & Schmidhuber [24] in 1997. They solve the problem of vanishing and exploding gradient problem that is prevalent in a simple recurrent structure,as it allows some states to pass without activation. Along with sequential input and hidden state input they maintain a cell state which helps to model memory in long sequences. In a LSTM h_{t-1}(hidden state) and C_{t-1}(cell state) the next hidden state is computed using a series of computation described below.

$$f_t = \sigma(W_f \bullet [h_{t-1}, x_t] + b_f) \tag{1}$$
$$i_t = \sigma(W_i \bullet [h_{t-1}, x_t] + b_i) \tag{2}$$
$$\tilde{C}_t = \tanh(W_c \bullet [h_{t-1}, x_t] + b_c) \tag{3}$$
$$C_t = f_t \bullet C_{t-1} + i_t \bullet \tilde{C}_t \tag{4}$$
$$o_t = \sigma(W_o \bullet [h_{t-1}, x_t] + b_o) \tag{5}$$
$$h_t = o_t \bullet \tanh(C_t) \tag{6}$$

The encoder is created using pair of LSTM running in opposite direction, then these states are concatenated and multiplied with their attention weights to generate the final context vector.Basically the weights matrices in the equations above are the parameters that are learned during training process.

$$c_t = \sum_{i=1...|s|} \frac{h_t^T W_b b_i}{\sum_j (h_t^T W_b b_j)} [\overrightarrow{b_i}; \overleftarrow{b_i}]$$

Similar to the technique used in [17] initialisation of the hidden states of decoder is done with concatenation of final state of forward and backward pass of encoder.

5.1.2. GRU

Gated Recurrent networks were developed by Cho et al. [21] in effort to design recurrent encoder decoder architecture in 2014. They are relatively simpler than LSTM and retain majority of its advantages. They have two gates instead of three in LSTM and the logical reset gate is distributed in r_t and z_t, thus even GRU dont suffer from vanish or exploding gradient issue. But rather with lesser parameters they are more easily trained. Below is described the computation for next hidden state.

$$z_t = \sigma(W_z \bullet [h_{t-1}, x_t] + b_z) \tag{7}$$
$$r_t = \sigma(W_r \bullet [h_{t-1}, x_t] + b_t) \tag{8}$$
$$\tilde{h}_t = \tanh(W \bullet [r_t * h_{t-1}, x_t]) \tag{9}$$
$$h_t = (1 - z_t) \bullet h_{t-1} + z_t \bullet \tilde{h}_t \tag{10}$$

Making a encoder using GRU is inherently similar to a LSTM, rather its the underlying architecture difference of recurrent layer which would create difference in learning and optimal decent.

5.1.3. CNN

Convolutional encoder are extension to pooling encoder proposed by Ranzato et al.[25], although it does not contain any pooling/down-sampling layers. The encoder outputs a vector of fixed length but the width is dependent upon the kernel size used. Residual Connections as described in [26] are used where input and output of the convolution layer are further passed by a non-linearity.

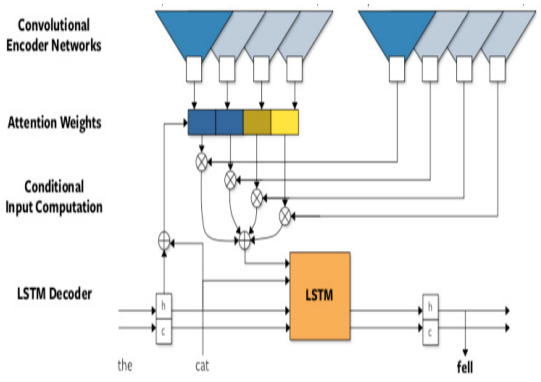

Fig. 2. Left network in top layer is CNN_1 and other on right is CNN_2

The primary difference in CNN encoder is the architecture has two separate stacked convolution networks as shown in Fig. 2[1]. The CNN_1 computes h_j which is used to compute attention scores a_i and the input to decoder c_i is computed by CNN_2 and attention weights.

$$c_i = \sum_{i=1}^{m} a_{ij} * CNN_2(e)_j$$

In the equation above the CNN_2 operate on the word embedding of the input sequence which are represented by e and in the summation m is the dimensionality of embedding space.

5.2. Decoder

A standardized decoder[22] with each of described encoders in section-5.1 was used inexperiments. Taking the mathematical formulation described in section-3, the decoder aims to maximize the conditional probability on a word level.

$$P(q|s) = \prod_{i=1}^{|q|} softmax(W_s tanh(W_i \bullet [h_i; c_i] + b_i))$$

In the above equation the product term represents word output probability $P(q_i|s, q_{<i})$. It is made using LSTM units and the probability of each generate word is dependent upon hidden state and attention weight. The hidden state(h_t) is computed by input of previous hidden state(h_{t-1}) and previous output.

6. Experiments

In the experiments multiple parameters are tested and reasons for the effects are addressed. All models are tested with varying dimensionality of the word embeddings, it was observed the model performs better on evaluation measure when pre-trained word vectors are used in comparison to random trainable embeddings, for pre trained vectors Stanford's Glove vectors [28] were used, its effect with dimensionality is shown below in Fig. 3, the other of the model characteristic used study this variation are discussed further.

For training the model Stochastic gradient decent(SGD) was used to minimize over the objective function, which has been modelled as negative of logarithmic likelihood of training samples over network parameters.In all the model were trained for 10 epochs with a learning rate of unity, which was decayed(by half of 0.5) if validation loss was increased, this attained good validation loss with SGD for recurrent as well as convolutional encoders[2].The GPU used for the experiments was Nvidia GeForce GTX 745 having a 4 Gb of memory according to which the batch size was kept at the value of 48.

In the further experiments word vectors of the size 300 were used as they maximize context and effects of other parameters on the architectures can be observed. With recurrent and convolution layers dropout layers with value 0.3 were added to avoid over-fitting and assist better generalization and in our decoder beam size of 5 was used characterising to the path splits before decoder predicts a word. The co-relation of model complexity and BLEU-1 was observed, for which size of recurrent units for LSTM and GRU and layers of convolutions in case of CNN encoder were varied and the results can be seen in Fig. 4 and Fig. 5 respectively.

[1] Image taken from [27] under BSD License

[2] These would differ if choice of optimizer is changed to Ada-Grad or Adam

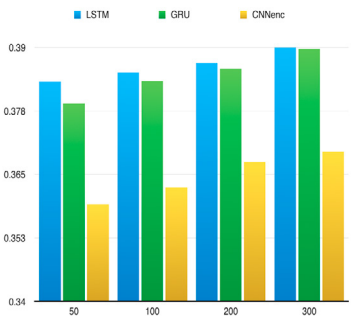

Fig. 3. BLEU-1 variation with embedding size

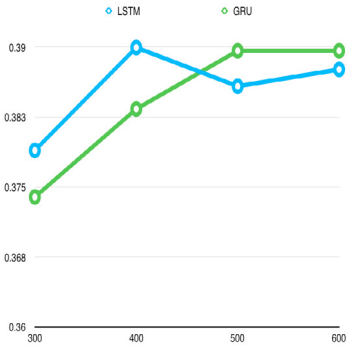

Fig. 4. BLEU-1 variation with size of recurrent units in LSTM and GRU based encoders

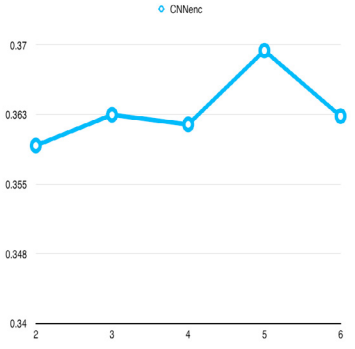

Fig. 5. BLEU-1 variation with convolution layers in CNN encoder

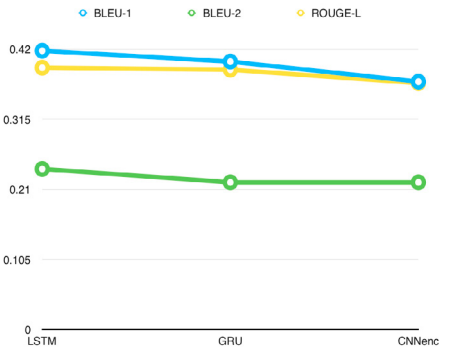

Fig. 6. Evaluation Results

7. Results and Analysis

For evaluation of the generated question against the original question sequence two measures BLEU and ROGUE have been used.

BLEU[29] originally created to measure quality of machine translation with respect to human translation. It calculates a N-gram precision between the two sequence and also imposes commensurate penalty for machine sequence being shorter than human one. Here BLEU-1 and BLEU-2 corresponding to uni-gram and bi-gram versions of the approach is used for evaluation.

The other measure being used is ROGUE-L[30], which is dependent on the length of longest common sub-sequence (LCS) with strictly increasing indices. It considers fractional length of LCS over sequence length as precision/recall with respect to one and vice-versa for other, and the final is calculated as F-measure of these values above.

Table 2. Training time for models

Model	Time(seconds)
LSTM	13,599.88
GRU	11,819.26
CNN_{enc}	10,304.63

Now further the comparative results models using the three previously described encoders in Fig. 6 has been shown correspondingly the Table 2 shows the training time for each of the model. Hence it can be seen, GRUs achieve evaluation scores close to the LSTM model with little lesser training time whereas CNN models have scores lesser

than both recurrent structures but achieve performance which is competitive given it much speed, which can be useful in application which train on large scale corpuses.

8. Conclusion

This study explores many of the existing techniques used with encoder-decoder architectures and their application for generating question sequences from text. Firstly the notion of using GRU as a viable alternative to LSTM has been established by experimenting with size of recurrent units as depicted in Fig. 4, although GRU requires more units to attain similar performance but have training time ≈15% lesser than the best performing LSTM. The application of convolutional encoder for the task of question generation has been described, which achieves evaluation scores remarkably close to currently state of the art systems with ≈25% lesser training time, this can be explained not only by fast computation of convolution layer than over recurrent architectures, but also its effective generation of text representation which captures semantics and achieves competitive evaluations.

In future we aim to develop a question generation system which can form question directly from input text without having to rely upon reference answer text leading to more usable systems. We hope to achieve it by applying ontological techniques to paragraph text, to extract candidate sentences and further using neural network based approaches to form questions.

Acknowledgements

This research has been pursued at WISoC Lab with support from Department of Computer Science at Birla Institute of Technology and Science, Pilani. We are thankful to the authorities for providing us the infrastructure facilities and opportunity to undertake this research project.

References

[1] Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250.

[2] Brown, J. C., Frishkoff, G. A., & Eskenazi, M. (2005, October). Automatic question generation for vocabulary assessment. In Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing (pp. 819-826). Association for Computational Linguistics.

[3] Liu, M., Calvo, R. A., & Rus, V. (2012). G-Asks: An intelligent automatic question generation system for academic writing support. Dialogue Discourse, 3(2), 101-124.

[4] Mitkov, R., & Ha, L. A. (2003, May). Computer-aided generation of multiple-choice tests. In Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing-Volume 2 (pp. 17-22). Association for Computational Linguistics.

[5] Kunichika, H., Katayama, T., Hirashima, T., & Takeuchi, A. (2004). Automated question generation methods for intelligent English learning systems and its evaluation. In Proc. of ICCE.

[6] Weizenbaum, J. (1966). ELIZA—a computer program for the study of natural language communication between man and machine. Communications of the ACM, 9(1), 36-45.

[7] Rus, V., Wyse, B., Piwek, P., Lintean, M., Stoyanchev, S., Moldovan, C. (2010, July). The first question generation shared task evaluation challenge. In Proceedings of the 6th International Natural Language Generation Conference (pp. 251-257). Association for Computational Linguistics.

[8] Heilman, M., & Smith, N. A. (2010, June). Good question! statistical ranking for question generation. In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (pp. 609-617). Association for Computational Linguistics.

[9] Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., & Deng, L. (2016). Ms marco: A human generated machine reading comprehension dataset. arXiv preprint arXiv:1611.09268.

[10] Miller, G. A. (1995). WordNet: a lexical database for English. Communications of the ACM, 38(11), 39-41.

[11] Agarwal, M., Mannem, P. (2011, June). Automatic gap-fill question generation from text books. In Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications (pp. 56-64). Association for Computational Linguistics.

[12] Becker, L., Basu, S., & Vanderwende, L. (2012, June). Mind the gap: learning to choose gaps for question generation. In Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 742-751). Association for Computational Linguistics.

[13] Mazidi, K., Nielsen, R. D. (2014). Linguistic considerations in automatic question generation. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (Vol. 2, pp. 321-326).

[14] Labutov, I., Basu, S., & Vanderwende, L. (2015). Deep questions without deep understanding. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers) (Vol. 1, pp. 889-898).
[15] Lindberg, D., Popowich, F., Nesbit, J., & Winne, P. (2013). Generating natural language questions to support learning on-line. In Proceedings of the 14th European Workshop on Natural Language Generation (pp. 105-114).
[16] Mostow, J., Chen, W. (2009, July). Generating Instruction Automatically for the Reading Strategy of Self-Questioning. In AIED (pp. 465-472).
[17] Du, X., Shao, J., & Cardie, C. (2017). Learning to ask: Neural question generation for reading comprehension. arXiv preprint arXiv:1705.00106.
[18] Yang, Z., Hu, J., Salakhutdinov, R., & Cohen, W. W. (2017). Semi-supervised QA with generative domain-adaptive nets. arXiv preprint arXiv:1702.02206.
[19] Wang, T., Yuan, X., & Trischler, A. (2017). A Joint Model for Question Answering and Question Generation. arXiv preprint arXiv:1706.01450.
[20] Du, X., & Cardie, C. (2017). Identifying where to focus in reading comprehension for neural question generation. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (pp. 2067-2073).
[21] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.
[22] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Advances in neural information processing systems (pp. 3104-3112).
[23] Luong, M. T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. arXiv preprint arXiv:1508.04025.
[24] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.
[25] Ranzato, M. A., Chopra, S., Auli, M., & Zaremba, W. (2015). Sequence level training with recurrent neural networks. arXiv preprint arXiv:1511.06732.
[26] Gehring, J., Auli, M., Grangier, D., & Dauphin, Y. N. (2016). A convolutional encoder model for neural machine translation. arXiv preprint arXiv:1611.02344.
[27] https://github.com/facebookresearch/fairseq
[28] Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).
[29] Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002, July). BLEU: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting on association for computational linguistics (pp. 311-318). Association for Computational Linguistics.
[30] Lin, C. Y., & Och, F. J. (2004, July). Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics (p. 605). Association for Computational Linguistics.